

each bit defines an error correcting capability based on whether the bit is set to 1 or 0.  
The meaning of each bit is provided in Table 6.

Table 6	
Byte Bit Position	Meaning
Bit 0	Other
Bit 1	Unknown
Bit 2	None
Bit 3	Single Bit Error Correcting
Bit 4	Double Bit Error Correcting
Bit 5	Error Scrubbing

```

5      dw    BIT_FIELD_ID          ; Data identifier
      db    "Error Correcting Capability",0 ; Field description
      db    1                      ; Bytes used for bit-field
      db    "No",0                 ; String to use when bit = 0
      db    "Yes",0               ; String to use when bit = 1
10     ; Begin bit field encoding for this value
      db    00h, "Other",0
      db    01h, "Unknown",0
      db    02h, "None",0
      db    03h, "Single Bit Error Correcting",0
15     db    04h, "Double Bit Error Correcting",0
      db    05h, "Error Scrubbing",0
      dw    END_OF_BIT_FIELD_ID

```

As illustrated, the remaining two bits of the bit field are not defined, and as such,  
the END\_OF\_BIT\_FIELD\_ID key is used to end the structure definition, such that the  
remaining two bits do have to be defined in the template file.

#### D. ENUM\_ID and END\_OF\_ENUM\_ID

This key defines an enumerated value, meaning that the field's numerical value  
represents some type of defined setting. Enumerated values are always one byte.

##### Encoded Value:

```

ENUM_ID          0xFFFF4
END_OF_ENUM_ID  0xFFFF5

```

Format:

5     dw    ENUM\_ID  
      db    Null terminated string for field title  
      db    xx – Enumerated value setting  
      db    Null terminated string describing the previous enumerated value  
      ...repeat this section until all values are defined...  
      dw    ENDOF\_ENUM\_ID

Notes:

10       The ENDOF\_ENUM\_ID key indicates the end of the enumerated values so that  
      undefined values do not have to be defined in the template file. For example, an  
      enumerated value stored in the SMBIOS database is on byte long or 8 bits. That means  
      that there are 2<sup>8</sup> possible values. If less than all values are currently defined, the  
      ENDOF\_ENUM\_ID may be used after the last defined value.

15

Assembler Data Example:

      Illustrated below is the structure definition for data related to the processor type,  
      which is Type 4, offset 05h. The enumerated value for the offset has 6 possible values,  
20       with each value of the enumerated value indicating a different type of processor. Table 7  
      provides the processor designation for each byte value of the enumerate value.

Table 7	
Byte Value	Meaning
01h	Other
02h	Unknown
03h	Central Processor
04h	Math Processor
05h	DSP Processor
06h	Video Processor

25       dw    ENUM\_ID                                 ; Data identifier  
      db    "Processor Type",0                     ; Field Description  
      db    01h, "Other",0  
      db    02h, "Unknown",0  
      db    03h, "Central Processor",0  
      db    04h, "Math Processor",0  
      db    05h, "DSP Processor",0  
30       db    06h, "Video Processor",0  
      dw    ENDOF\_ENUM\_ID

### **E. GROUPED\_BIT\_FIELD\_ID, GBF\_END\_GROUP, and ENDOF\_GROUPED\_BIT\_FIELD\_ID**

These descriptor keys define a grouped bit field. A grouped bit field differs from a normal bit field in that the groups of bits collectively represent some sort of setting of the computing system similar to an enumerated value. For example, a grouped bit field having a size of 1 byte may have bits 0-2 representing 8 different settings based on their value, while bits 3 and 4 represent 4 other settings, etc. In this instance, the GROUPED\_BIT\_FIELD\_ID descriptor key represents the beginning of the grouped bit field. The GBF\_END\_GROUP descriptor key indicates the end of the definitions for each group within the grouped bit field. Similar to an enumerated value, each group within the grouped bit field contains description strings for each possible value of the bit group. Finally, since the number of possible values per group is defined as  $2^n$  (where n is the number of bits in the group), all possible values may not be defined. As such, the ENDOF\_GROUPED\_BIT\_FIELD\_ID indicates the end of the grouped bit definitions, so that undefined groups of bits do not have to be defined in the template file. For example, if the byte size is 1, but only one bit group from bits 3:0 is defined, the remaining bit groups do not have to be defined in the template file.

#### Encoded Value:

GROUPED_BIT_FIELD_ID	0xFFFF6
GBF_END_GROUP	0xFFFF7
ENDOF_GROUPED_BIT_FIELD_ID	0xFFFF8

#### Format

dw	GROUPED_BIT_FIELD_ID
db	Null terminated string for field title
db	Number of bytes occupied by the grouped bit field
→	(Begin bit field group encoding)
db	Group start bit
db	Group end bit
db	Null terminated string of group description
→	db Bit group value
	db Null terminated string describing the above value
	... repeat this section for each possible value in the group ...
db	GBF_END_GROUP
→	... repeat this section for all groups ...
dw	ENDOF_GROUPED_BIT_FIELD_ID